**Normalization**

Normalization of columns means adjusting the values in one or more columns so that they fall within a certain standardized representation, often using **0s and 1s**. This process is very important in **machine learning**, because raw data usually contains categorical features such as text, country names, genres, and ratings that models cannot directly understand. By normalizing these columns, we can transform them into a structured numerical format that makes them compatible with algorithms and improves model accuracy.

* The **type column** contained only two categories:

*Movie* and *TV Show*. Since it is binary, we can easily normalize it using **label encoding** where one category is represented as 0 and the other as 1.

df = df.withColumn('movie', array\_contains(col('type'), 'Movie'))

df = df.withColumn('tv\_show', array\_contains(col('type'), 'TV Show'))

df = df.withColumn("movie", when(col("movie") == True, 1).otherwise(0)) \

.withColumn("tv\_show", when(col("tv\_show") == True, 1).otherwise(0))

The **country column** contained more than 100 unique country names. To minimize complexity, we used **OneHotEncoder**. This converts country names into indexed values and then encodes them into 0 and 1 columns.

df = df.withColumn(

'country',

split(col('country'), ' ')

)

df = df.withColumn('country', expr("country[0]"))

df = df.replace(

{

'United': 'USA',

'Not': 'Not Found',

'South': 'South Africa',

'Hong':'Hong Kong',

'New': 'New Jersey',

'West': 'West Indies',

'North': 'North Korea',

'Central': 'Central African Republic',

'East': 'East Timor'

},

subset=['country']

)

The **listed\_in column** (genres) had around 10–20 categories like *Drama, Comedy, Action, Horror*. This required **splitting the string column into arrays**, then using **explode** to create multiple rows, followed by encoding.

for cat in categories:

col\_name = cat.strip().replace(" ", "\_").replace("&", "and").replace(".", "").replace("-", "\_").replace("...", "etc").replace("/", "\_").replace("'", "").replace(",", "").replace(" ", " ")

df = df.withColumn(col\_name, when(col(col\_name) == True, 1).otherwise(0))

The **rating column** had 10–15 unique values such as *PG, R, TV-MA, TV-Y*. We applied a similar process (StringIndexer + OneHotEncoder) to normalize ratings.

for rating in distinct\_ratings:

col\_name = rating.strip().replace(" ", "\_").replace("-", "\_").replace(".", "").replace("/", "\_").replace("&", "and").replace("'", "")

df = df.withColumn(col\_name, when(col(col\_name) == True, 1).otherwise(0))